[0042] The indexed subsequences can be sorted as shown in **Figure 3A**. The sort can be by nucleotide. The sort can be used to create an index where identical subsequences are adjacent to one another in the index so that read pairs having subsequences in common readily can be identified. For example, **Figure 3A** shows us that subsequence AGCC is common to reads 3 and 61, and subsequence CCCT is to reads 3, 50 and 61.

Please replace paragraph [0043] with the following amended paragraph:

[0043] The method can further include the step of discarding read subsequences having more than a cutoff number of occurrences, from the indexed subsequences. This step can be used, *e.g.*, to remove from the index subsequences that are repeated so many times that it is not efficient to include them for the purposes of detecting read overlaps. For example, in the human genome, the nucleotide sequence for the amino acid Alu is repeated as many as a million times throughout the genome, and so it is not efficient to consider the nucleotide codi[[~~gn~~]]ng for Alu subsequences when attempting to detect read overlaps. The cutoff number can be determined by the initial redundancy of sequencing that was used. For instance, if the total length of the reads is 10L, where L is the length of the genome in nucleotides, the initial redundancy of sequencing is 10, and therefore an average region of the genome is expected to be represented 10 times by reads. Therefore, a subsequence that occurs considerably more than 10 times is likely to be repetitive. A subsequence that occurs 100 times is likely to be so repetitive that it is not efficient to include it for the purposes of detecting overlaps, and such subsequences can be discarded. The best cutoff number is the largest cutoff number that allows for an acceptably low number of pairwise comparisons.

Please replace paragraph [0044] with the following amended paragraph:

[0044] A Table of Sorted Sequences, such as that depicted in **Figure 3A**, can also include a "Remove Flag" column (not shown) that identifies subsequences that occur more than a cutoff number of times. This step can be used to remove highly repetitive subsequences

from consideration to increase the efficiency of the merging step, since these subsequences likely will falsely indicate overlap between the reads in which they appear

Please replace paragraph [0047] with the following amended paragraph:

[0047] Figure 3B is an illustration of an exemplary method of extracting read pairs having common subsequences by building a Table of Read Pairs with Common Subsequences from a sorted Table of Sorted Subsequences in accordance with the present invention. The Table of Sorted Subsequences is a sorted index of the Table of Subsequences depicted in **Figure 2**. The Table of Read Pairs with Common Subsequences is generated from the Table of Sorted Sequences by extracting a list of read pairs having at least 1 subsequence in common. Each read pair is associated with the number of common subsequences indicated in parenthesis.

Please replace paragraph [0048] with the following amended paragraph:

[0048] Figure 3B illustrates that this method of indexing the subsequences and extracting the read pairs having common subsequences significantly reduces the number of read comparisons for the merge from the number of reads squared (N^2), to the number of read pairs that have one or more common subsequences. For example, using the method of the present invention, read 1 will be compared with reads 3 and 50, but not with 61 or 14, whereas with an N^2 comparison method, read 1 would be compared with every other read, including 14 or 61. Similarly, because read 105 has no subsequences in common with reads 1, 3, 14, 50 and 61, these comparisons are not made. This method can include sorting the table of subsequences by nucleotide.

Please replace paragraph [0050] with the following amended paragraph:

[0050] Figure 4 is an illustration of an exemplary method of merging read pairs along a continuum. This merge can be performed by choosing one read pair, *e.g.*, read pair (3,61) from **Figure 3B**. A direction is arbitrarily chosen for the first alignment, in this

case, it is decided to represent the reads from left to right. Pair (3,61) is aligned along a continuum running from left to right. Next, a read pair can be chosen that includes one of the reads in the pair already aligned. Pair (3,50) can be chosen, *e.g.*, and read 50 aligned with read 3 in **Figure 4**. The merge can be built pair by pair as shown in **Figure 4**. It should be noted that read 14, shown in **Figure 4**, would not be aligned from the data shown in **Figure 3<u>B</u>** because it does not share a common subsequence with the other reads shown. However, as the reads are merged with data not shown in **Figure 4**, other pairs can be aligned until a read is added that has a subsequence in common with read **14**, and read **14** would be merged. From the merged reads, the resolved sequence ATAGCCCTGCGCCTATCG, indicated below the continuum line in **Figure 4**, can be obtained.

Please replace paragraph [0051] with the following amended paragraph:

[0051] Alignments optionally can use the associated position of the subsequences on the reads to confirm overlap. For example, consider the two sequences: GATCCCATGCGCA and ATAGCCCTATGAT. These sequences share common subsequences GAT and CCC. We know from the associated position information that CCC begins at base 4 for the former and base 5 for the latter, and the GAT subsequence begins at base 1 for the former and base 11 for the latter. This position information indicates that there is no overlap between these two sequences since the position of the common subsequences does not allow for alignment. Consider now the first sequence above and the sequence CCCATGCGCATAT. We know that the common subsequence CCC begins at base 4 for the former and base 1 for the latter, and the common subsequence <u>[[~~CGC~~]]GCG</u> begins at base 9 for the former and base 6 for the latter. This position information indicates that there is overlap between these two sequences. Comparing the rest of the intervening subsequences confirms the overlapping region CCCATGCGCA.

<u>In the Claims</u>

Please amend claims 1-21 to read as follows.